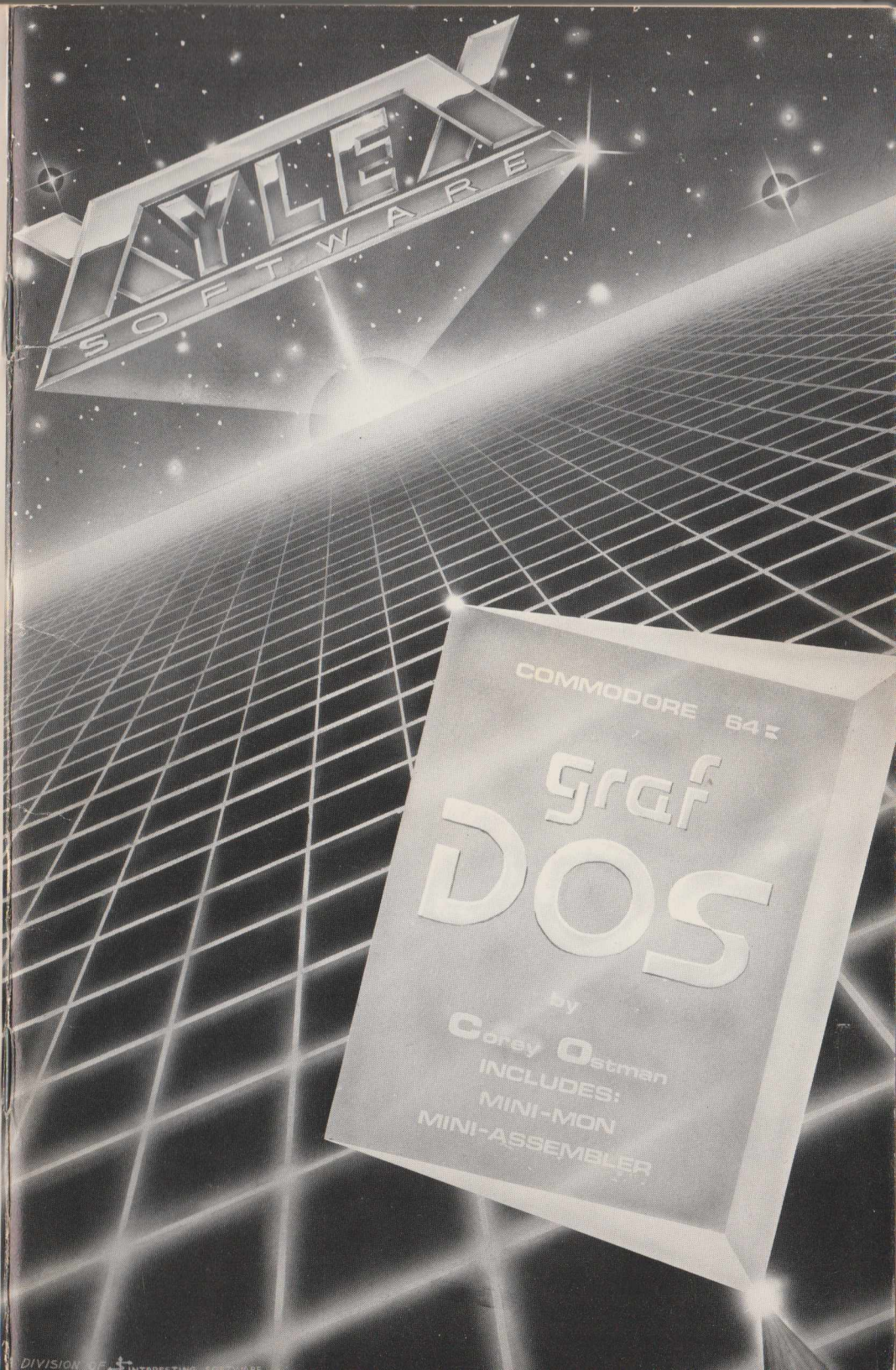
SOFTWARE
COMMODORE 64
graf
DOS
by
Corey Ostman
INCLUDES:
MINI-MON
MINI-ASSEMBLER

NOTICE

We reserve the right to make changes and improvements in the product described in this manual at any time and without notice.

LIMITATION OF WARRANTIES AND LIABILITY

LIMITED WARRANTY ON MEDIA. XYLEX and Interesting Software warrants the disk on which the software is recorded to be free from defects in materials and faulty workmanship under normal use for a period of 90 days after the date of original purchase. If during this 90-day period a defect in the disk should occur, the disk may be returned to XYLEX Software for replacement without charge, provided that you have completed the enclosed registration form and returned it to XYLEX Software. Your sole remedy in the event of a defect in a disk is limited to replacement of the disk as provided above.

LIMITATION ON WARRANTY AND LIABILITY

EXCEPT AS EXPRESSLY PROVIDED ABOVED FOR MEDIA , XYLEX SOFTWARE, INTERESTING SOFTWARE, MAKE NO WARRANTY OF ANY KIND WITH REGARD TO THIS MATERIAL, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THEY SHALL NOT BE LIABLE FOR ERRORS CONTAINED HEREIN OR FOR INCIDENTAL OR CONSEQUENTIAL DAMAGES IN CONNECTION WITH THE FURNISHING, PERFORMANCE, OR USE OF THIS MATERIAL.

COPYRIGHT NOTICE

INTRODUCTION TO grafDOS

Congratulations! You have just purchased one of the most valuable programs for your Commodore 64. grafDOS was designed to both overcome some of the limitations of Commodore DOS and enhance the BASIC programming language primarily in the graphics area. In general, to make life easier.

grafDOS will allow you to access the disk with simple, easy to use commands without having to constantly refer to the manual. It is different from the DOS wedge in that it is a completely new operating system with its own set of commands. The DOS wedge is merely a form of shorthand for existing Commodore commands, (although it does have some enhancements).

In addition to Commodore's BASIC, grafDOS provides an extensive set of commands for manipulating low resolution graphics, high resolution graphics, character graphics and sprites. It also provides easier manipulation of text and error trapping.

As an added bonus, the grafDOS Master disk includes a powerful machine language monitor that will allow the user to disassemble code, examine memory in both ASCII or hex, edit memory, move memory and much much more. The monitor will also allow you to create machine language programs through it Mini-Assembler. This is a stand-alone program and does not need grafDOS to operate.

The grafDOS Master disk comes completely COPYABLE and we encourage you to make backup copies FOR YOURSELF. It is our intent to make this product as useful as possible by allowing you the ability to make copies. However, we ask you not to abuse our intentions by making copies for your friends or for other purposes. By doing so only encourages the unfortunate situation of more software copyprotection and thereby making programs less useful. We hope that the commercial success of non-protected software such as this one will encourage other companies to do the same.

However, the author has spent over a year developing this program and we consider this one of the most useful products for the Commodore 64 computer. We will act to protect our copyrights of this manual and this program and do whatever is necessary to protect our legal rights. We have several ways of tracing copies and we will use them if necessary. Please act conscientiously and remember, we are trying to stay in business by bringing you useful software packages.

Although we feel this operating system is nearly perfect, there is always room for future improvements. As a valued customer, your opinion is very important to us. Any comments or criticisms you may have will be greatly appreciated, so drop us a letter or give us a call.

Be sure to fill out the warranty card and mail it in so we may keep you posted on future developments and put you on our mailing list.

grafDOS BASIC TABLE of CONTENTS

CHAPTER 5
THE MACHINE LANGUAGE MONITOR

APPENDICES

CHAPTER 1-GETTING STARTED

BACKGROUND

Learning to use grafDOS and its operating system involves learning some new instructions, many of which are easily learned extensions of Commodore BASIC. This manual is not a BASIC tutorial and assumes that the user is already able to write simple BASIC programs.

Before using the Commodore 64 and its disk drive, make sure that you are well acquainted with the operating manuals for both of these units. This will insure your being able to use grafDOS to its full extent.

BOOTING grafDOS

To start up grafDOS BASIC, you must first load it into the Commodore 64, allowing its commands to be included with those already present in the resident BASIC and DOS.

Turn on your Commodore 64 system and insert the grafDOS SYSTEM MASTER disk.

Make sure the label side is up when inserting the disk. You will feel some slight pressure and hear a click. Close the disk drive door.

The introductory screen of the Commodore 64 should appear. The READY prompt indicates you are now in Commodore BASIC.

Next, you must transfer grafDOS from the disk into the memory of the computer. This is accomplished very simply by typing:

```
LOAD "GRAFDOS",8  <ret>
```

The disk will make a "whirring" sound, and Commodore BASIC will inform you that it is LOADing the program.

The BASIC prompt 'READY' will reappear, now type:

```
RUN  <ret>
```

The grafDOS title page will appear on the screen, and the disk will commence loading grafDOS from disk. After a few moments, the following title page will appear:

```
**** COMMODORE 64 GRAFDOS BASIC ****
    COPYRIGHT (C) 1982 BY XYLEX

OK
```

...and the flashing cursor will appear below this message.

Type in the following command:

NEW <ret>

This will clear any "garbage" that was in memory at the time the Commodore 64 was turned on.

*** YOU ARE NOW OPERATING IN grafDOS BASIC!!! ***

IF BOOTING FAILS

If you cannot get grafDOS to successfully load, try re-reading the BOOTING section. Start from scratch and try again. This will cure your problem 99% of the time!

Although it isn't likely, if for some reason the grafDOS MASTER DISK has suffered damage, please return the entire package to the place of puchase for a replacement.

INITIALIZING DISKETTES WITH grafDOS

The grafDOS MASTER DISK that was included with this manual is a very special disk and should be safely guarded. This is why the "write protect" notch has been covered on the MASTER DISK, preventing you from inadvertently destroying the grafDOS system.

Since you are already in grafDOS, remove your grafDOS disk from the drive, replacing it with a "blank" disk, a disk that doesn't have any programs on it. This disk must be initialized (or formatted) before you can use the disk for storage.

Make sure the blank disk is in the disk drive. If it is, type the following:

INIT "MY DISK" <ret>

The computer will respond with:

ARE YOU SURE (Y/N)?

If you're sure, type Y, and the computer will initialize the blank disk. By the way, the name MY DISK is just simply a sample name. Any other name could be used. This will allow you to title the disk for future reference.

After a short delay, the BASIC prompt 'OK' will reappear. The disk should now be initialized.

LOADing, SAVEing & RUNning with grafDOS

LOADing, SAVEing and RUNning programs using grafDOS are just as simple and in fact simpler than using standard Commodore DOS. For example, under standard DOS you would have to LOAD a file by typing LOAD "A FILE",8. Under grafDOS, you just have to enter LOAD "A FILE". The same is true with all of the grafDOS commands.

For a first try at using grafDOS, type in this simple program:

```
100 HOME
110 VTAB 10
120 SOUND 17,75 @ 200
130 PRINT "HI! I'M IN GRAFDOS BASIC..."
140 SOUND 34,75 @ 200
150 FOR G=1 TO 500:NEXT G
160 HOME
170 END
```

Now, let's see what this program does...

type RUN <ret>.

The screen will clear, you will hear a tone, and HI! I'M IN GRAFDOS will appear. Another tone will be played, the screen will clear once more and the OK prompt will return.

To store this program on disk, type the following:

SAVE "FIRST PROGRAM" <ret>

When you type this command, the disk will whir for a few seconds, and the program will then be saved on disk!

To see that the program has indeed been saved. Type in the command NEW to clear away the program. Then type:

LOAD "FIRST PROGRAM" <ret>

Type RUN (followed by <ret>) and your program will execute once more. Actually, instead of having to type LOAD and then RUN, you could have just typed:

RUN "FIRST PROGRAM" <ret>

...and the program would have loaded and run in one step.

CATALOG, the disk directory

You have already stored one program on your disk. To see which programs are on your disk (there should be only one), type:

```
CATALOG <ret>
```

And the following will be printed:

```
DISKETTE DIRECTORY: MY DISK

FILE NAME           TYP   BLOCKS
---------           ---   ------
FIRST PROGRAM       PRG   1

OK
```

You can now see FIRST PROGRAM listed under FILE NAME. This is followed by PRG (PRoGram) showing its file TYPe. The 1 stands for the number of blocks (sectors) the program uses for storage. This is a relatively short program. That's why it only takes up 1 block.

* Note - if there are more files on your diskette than can be displayed on the screen at one time, the directory will pause after the screen has filled. To see the rest of the directory, simply press the space bar.

This command is different from Commodore's directory command in that it will leave any BASIC program in memory undisturbed. You will find this one of the most useful commands in grafDOS.

RECOVERING FROM ACCIDENTAL RESETS

If by accident, you should type RUNSTOP/RESTORE, the computer will reset itself and go back to the standard Commodore 64 video display. You may return to grafDOS BASIC by typing the following command:

```
BASIC <ret>
```

The normal grafDOS screen will return, and your program will still be intact.

*** By now, you should be relatively comfortable using these DOS commands. Throughout this chapter, we have reminded you to type a <ret> after each command, however we will now assume that it is an involuntary action and it will not be necessary for us to remind you.

CHAPTER 2-MORE DOS COMMANDS

In chapter 1, we covered some of the essential DOS commands to get you started. Here we will cover the rest of the available DOS commands under grafDOS.

RENAMEing FILES

For one reason or another, you may need to change the name of one of your programs. Let's suppose that you're tired of calling your first program "FIRST PROGRAM", and you'd like to call it "SOUNDS". To accomplish this, type the following:

```
RENAME "FIRST PROGRAM","SOUNDS"
```

...Then type CATALOG to see for yourself that the filename was changed to "SOUND"!

DELETEing FILES

To remove unwanted files from your disk, use the DELETE command. For example, if you didn't want "SOUNDS" to be on your disk anymore, you would type:

```
DELETE "SOUNDS"
```

...and it would be removed from your disk. (Care should be taken when using the DELETE command as you could lose valuable programs by accidentally deleting them.)

UPDATE

This command UPDATEs your disk so that all free sectors are placed consecutively next to each other rather than being scattered all over the disk. Caution should be used when using this command. If there are any sequential files on the disk, they will be destroyed. It is a good idea to have a backup before using this command.

STAT "MEM" and STAT "SYS"

The STAT "MEM" command will perform a memory diagnostics to find any bad RAM. This procedure will take several minutes to complete.

The STAT "SYS" command will perform a system diagnostics making sure that everything is in operating order.

WATCH and OFF

WATCH will cause grafDOS to print to the screen to see what DOS commands are being executed. Conversely, OFF will disable the WATCH command.

USING grafDOS FROM WITHIN A BASIC PROGRAM

Often during the course of a BASIC program, its very useful to access DOS from within the program. For example, to get a catalog of the disk from within your BASIC program, you COULD NOT TYPE:

```
100 CATALOG
```

This is an illegal syntax and you would promptly receive a ?SYNTAX ERROR IN 100 upon execution. But, there is a way to access all the DOS commands from within BASIC, using the syntax:

```
100 PRINT CHR$(4);"command"
```

...where "command" is any valid grafDOS command.

Our example of wishing to catalog a disk from within a BASIC program would be accomplished in the following manner.

```
100 D$=CHR$(4)
110 PRINT D$;"CATALOG"
```

*** Note. The assignment of CHR$(4) to D$ in line 100 saves keystrokes when a number of DOS commands are used within a BASIC program.

If you RUN this program, the diskette catalog would be printed just as if you had typed CATALOG yourself! The CHR$(4) is the key to what's happening here. It tells grafDOS BASIC that the string after the PRINT statement is a DOS command and not a normal message that should appear on the video screen.

*** Exceptions to the rule

Quote marks around file names must be changed somewhat in using grafDOS commands within BASIC programs: For example, to RENAME two files we would have to use:

```
100 PRINT D$;"RENAME 'OLD','NEW'"
```

The apostrophe now takes the place of a quote. This is so we don't confuse BASIC with excess quote marks causing syntax. errors.

CHAPTER 3-USING MACHINE LANGUAGE FILES

BSAVE

In order to SAVE a section of memory, machine language program or not, the command BSAVE is used. The syntax for BSAVE is really quite simple. Let's say that you wish to save the current video screen under the name "B.VIDEO". The video screen occupies memory from 1024-2039. To SAVE this area, enter the following:

```
BSAVE "B.VIDEO",1024,2039
```

The current video screen is now saved on disk under the name "B.VIDEO". The use of the prefix 'B.' meaning BINARY in the file name helps the user to distinguish between machine language files (or sections of memory) and RUNnable programs.

BLOAD

This command is used to LOAD previously SAVEd machine language files or sections of memory. In our example, we SAVEd the video screen to disk, calling it "B.VIDEO". To make sure it was saved, enter the following:

```
BLOAD "B.VIDEO"
```

The previous video screen should now reappear.

*** NOTE: On newer COMMODORE 64s you may get a totally blank screen. If this is the case, you must poke the screen a different color to see the screen. (POKE 53281,6)

MOVING BETWEEN LANGUAGES: BASIC and the MINIMON

Suppose you've been using grafDOS BASIC, and you wish to enter the machine language monitor, MINIMON (included with grafDOS), because you want to enter some machine language routines. First, make sure the MINIMON has already been loaded (see MINIMON section of this manual). Now, to enter the MINIMON, type:

```
EXIT
```

If the MINIMON wasn't present in memory ($C000-CFFF), you would get the ?LANGUAGE NOT AVAILABLE ERROR...

But if the MINIMON was there, you have now jumped into the MINIMON machine language monitor program. Any BASIC program in memory will still be intact.

*** To learn more about MIMIMON, see chapter 5 of this manual.

CHAPTER 4-GETTING STARTED WITH grafDOS BASIC

BACKGROUND

As with the Disk Operating System, we assume that you are able to write simple BASIC programs. The graphics/sound commands of grafDOS BASIC are hopefully very simple to learn, and easily retained because of their straight-forward logic.

All commands operate in deffered-execution mode only. This means that they only work "inside" a BASIC program and cannot be entered after the OK prompt, or in the immediate mode.

LOW RESOLUTION GRAPHICS

The low resolution graphics screen allows you to program color graphics (in 16 colors) on a video screen divided into 40 columns by 25 rows of color blocks. Commands for this type of graphics are usually very easily understood, and should first be mastered BEFORE "graduating" to high resolution graphics.

To see your first sampling of low resolution graphics, type in this little program:

```
100 LGR
110 FOR X=0 TO 24
120 LCOL X
130 HLIN 0,39 @ X
140 NEXT X
150 FOR X=0 TO 39
160 LCOL X
170 VLIN 0,24 @ X
180 NEXT X
190 TEXT
```

Now, RUN it...
This program is very easily understood if we take some time to analyze each part:

100...LGR command clears the screen to low resolution graphics
120...LCOL sets the color for plotting
130...Draws a Horizontal LINe from (0,x) to (39,x)
170...Draws a Vertical LINe from (x,0) to (x,24)
190...Sets normal TEXT mode

The following program will demonstrate how versatile low resolution graphics can be, and will also reveal to you how easy "Pong" could be programmed!

```
100 POKE 53281,0:LGR
110 LCOL 7:HLIN 0,39 @ 0:HLIN 0,39 @ 24:VLIN 0,24 @ 0:
    VLIN 0,24 @ 39
120 X=10:Y=10
130 XA=1:YA=1
140 IF X>37 OR X<2 THEN XA=-XA:SOUND 34,75 @ 20
150 IF Y>22 OR Y<2 THEN YA=-YA:SOUND 16,75 @ 20
160 LCOL 6:LPLOT X,Y
170 FOR G=1 TO 10:NEXT G
180 LCOL 0:LPLOT X,Y
190 X=X+XA:Y=Y+YA
200 GOTO 140
```

HIGH RESOLUTION GRAPHICS

The high resolution graphics of grafDOS BASIC are almost as simple as the low resolution graphics, but take a little getting used to...

To see what we mean by HIGH RESOLUTION graphics, type in this short program...

```
100 HGR
110 FOR X=0 TO 255
120 Y=10*SIN(X*.1)
130 PLOT X,80-Y
140 NEXT X
150 KEY
160 TEXT
```

Here's an explanation of the program:

100...Set High resolution GRaphics
130...Plot sine wave on coordinate plane
150...Wait for KEYpress
160...Set to normal TEXT mode

For a complete look at all the grafDOS BASIC command set, consult the SUMMARY section of this manual.

PROGRAMMABLE CHARACTERS ON THE HIGH RESOLUTION SCREEN

One of the most unique features of grafDOS BASIC is that you are allowed to PLOT text, high resolution points, and programmable characters on the high resolution screen AT THE SAME TIME!

You have already seen how easy it is to PLOT points on the high resolution screen. Here we'll discuss how text and programmable characters are also plotted on the high resolution screen.

Four commands allow this to happen. The commands are:

...NOM, COPY, ALT, and WCHAR...

First, let's try writing text on the high resolution screen.

Type in this short program to illustrate how this is done:

```
100 HGR:                    REM CLEAR HIRES SCREEN
110 NOM:                    REM USE NORMAL CHARACTER SET
120 WCHAR 100,100 @ 1:  REM PLOT CHARACTER #1 AT (100,100)
130 WCHAR 110,100 @ 2:  REM PLOT CHARACTER #2 AT (110,100)
140 KEY:TEXT
150 END
```

After RUNning this program, you should see an 'A' and a 'B' appear on the hires screen...

By this time, two of the four commands should be clearer to understand:

WCHAR plots the character # on the hires screen.
NOM tells grafDOS BASIC that you'll be using the standard (normal) character set.

The character numbers range from 0 to 255 with A being character 1 and Z character 26. A full list can be found in the Commodore 64 Users Guide in Appendix E (pg 132).

If you wish to program your own characters, the command ALT, is used. This command specifies that grafDOS BASIC should use an alternate character set instead of Commodore's standard character set. This set will reside in memory starting at $9000 (the $ represents a hexadecimal number). For more information on programmable characters, please refer to the COMMODORE 64 REFERENCE GUIDE.

For your convenience, the grafDOS Master disk includes a character editor that will allow you to create your own character set or modify an existing one. Full instructions are given within the program.

Let's have a little fun. This short program will add a little "happy-face" to the character set, replacing the '@' symbol with a neat little smilin' face!

```
100 ALT                                  :REM USE ALTERNATE CHR. SET
110 COPY                                 :REM TRANSFER CBM CHR. SET
120 FOR X=0 TO 7                         :REM LINE 120 TO 150 CREATE
130 READ C:POKE 36864+X,C                :REM THE HAPPY FACE INTO
140 NEXT X                               :REM THE CHR. SET
150 DATA 60,66,165,129,165,66,60         :REM DATA FOR HAPPY FACE
160 HGR                                  :REM TURN ON HIRES GRAPHICS
170 WCHAR 0,0 @ 0                        :REM PLOT CHARACTER
180 KEY:TEXT                             :REM WAIT FOR KEYPRESS
```

Also, here's a little routine to plot hires characters on the screen. You set the coordinates using the variables X and Y. The string to be plotted is stored as the variable A$.

```
100 HGR                                  :REM TURN ON HIRES GRAPHICS
110 X=0:Y=0                              :REM INITIALIZE VARIABLES
120 A$="HI! I'M ON THE HIRES SCREEN!"    :REM CHARS. TO BE PLOTTED
130 GOSUB 1000                           :REM SUBROUTINE TO PLOT A$
140 KEY:TEXT                             :REM WAIT FOR KEYPRESS
999 END
1000 FOR L=1 TO LEN(A$):B=ASC(MID$(A$,L,1)):IFB>64THENB=B-64
1010 WCHAR X+(L-1)*8,Y @ B:NEXTL:RETURN
```

With this short routine (1000-1010) you can write strings on the hires screen. If you change '8' in 1010 to a smaller number the characters become packed. A larger number will spread them out. Replace '8' for some interesting effects.

In summary:

NOM	use normal Commodore character set
ALT	use alternate character set (at $9000)
COPY	copy Commodore character set into $9000
WCHAR	plot character on hires screen

SOUND SYNTHESIS

Although the Commodore 64 allows 3 voices to be played simultaneously, grafDOS BASIC only uses one of these voices to produce simple 'tones'. This command was designed to make sounds easier to do on the Commodore 64 for simple programming. Complex sounds can still be accessed through standard Commodore pokes as explained in the users manual.

The sound chip (SID) is accessed by the SOUND command. Through this command you are able to play a tone at a certain HI/LO frequency for a certain duration.

For example, to play the pitch C (octave #3), you would insert the following into your BASIC program:

```
100 SOUND 8,147 @ 200:REM PLAY C FOR DURATION 200
```

(see the MUSIC NOTE VALUES section of your Commodore 64 manual, appendix M on page 152.)

Those of you who want to change the attack/decay, waveform, and volume for grafDOS BASIC sounds. Use the following three simple POKEs:

```
POKE 35115,  attack/decay rate   (0-255)
POKE 35120,  waveform            (0-255)
POKE 35125,  volume              (0-255)
```

Here's a little program that will produce a rather well-known melody!

```
100 READ H,L,D:IF H=-1 THEN END
110 SOUND H,L @ D:GOTO 100
120 DATA 25,30,100,25,30,100,25,30,100,119,239,255,0,0,
         100,22,96,100,22,96,100
130 DATA 22,96,100,18,209,255,-1,-1,-1
```

*** NOTE: Some of the musical values will be different between the list found in the User's Guide and the Reference manual. You may use either one.

MISCELLANEOUS COMMANDS of grafDOS BASIC

The following commands allow the BASIC user to make programming easier.

HOME VTAB HTAB HIMEM CHAIN SPEED TRAP KEY TEXT

HOME...This command clears the video screen.

VTAB...This command moves the cursor to the given line number, for example: VTAB 10 would move the cursor to line #10, about the middle of the video screen. (min=0, max=24)

HTAB...Performs the same function as VTAB, but moves the cursor horizontally across the screen to the given column. (min=0, max=39)

HIMEM...Sets the BASIC highest memory to the given value. For example, HIMEM 8192 would set the highest portion of memory that BASIC could access to 8192. This command can be used for protecting the hires graphics page. Remember to reset it to 32767 or you may get an ?OUT OF MEMORY ERROR.

CHAIN...LOADs and RUNs other programs from within a BASIC program. This command does not transfer variable values.

SPEED...Sets the rate at which characters are outputed to the video screen. For example, SPEED 255 would delay a very long time before PRINTing the next character, whereas SPEED 0 would set the video printing speed to normal (fast).

TRAP...Tells grafDOS BASIC to jump to the given line number if an error occurs. An example of this would be: TRAP 1000, which would "jump" to line number 1000 if ANY error occurred from within the BASIC program itself.

KEY...Program execution halts until a key is pressed.

TEXT...Sets the video screen to normal text mode. This command should be used at the end of LGR and HGR routines.

The following program is an example of several commands of VTAB,HTAB, and HOME...

```
100 HOME
110 FOR X=20 TO 1 STEP -1
120 HTAB X:VTAB X
130 PRINT "HI THERE!"
140 NEXT X
```

*** NOTE: Further details can be found in the BASIC summary section in the appendix of this manual.

CONTROL/G (better known as "BEEP"?!)

If you have ever used or seen a TELETYPE, or know someone who uses one, you have probably heard of the infamous "bell".

This command causes the computer to beep, replacing the bell on teletypes.

The bell on grafDOS BASIC is accessed by pressing the CTRL key in conjunction with G. This can only be used when inside a PRINT statement (within the quotes). When the program PRINTs whatever is in the quotes and finds a CTRL G, it will beep. You may also store CTRL G's in a variable and PRINT the variable.

Another way is to PRINT CHR$(7), which is an equivalent command.

If you wish to change the pitch of the bell, do the following:

```
POKE 34507,  lo frequency    (0-255)
POKE 34512,  hi frequency    (0-255)
POKE 34517,  duration        (0-255)
```

Chapter 5 - COMMODORE 64 MINI-MONITOR

by Rob Beyer

I. GENERAL DESCRIPTION

The Commodore 64 Mini-Monitor is a general purpose monitor that allows the user to create, list, and alter assembly language programs. It has 20 commands that, disassemble memory, list memory in ASCII or hex, move memory, switch the kernal and BASIC to RAM, assemble using 6502 mneumonics, save machine code files, and load machine code files. Details on all the commands and how to use them are given in the next section. The program is located in the unused section of RAM at $C000 to $CFFF (note - the '$' indicates a hexadecimal number). The zero page ($00 - $FF) is $61 to $6D and $FB to $FD. These zero page locations should not interfere with any BASIC program in memory.

This manual is not a tutorial and is not designed to teach machine language. The user should have some familiarity with machine language, it's terminology, and how internal memory is structured. We recommend that you have a copy of the CBM-64 reference manual which will become invaluable as a programming aid on your computer. For those of you who are a little rusty (or just starting out) should read Chapter 5 on "BASIC to Machine Language". This should help you get started into fascinating world of machine language!

II. GETTING STARTED

1. Make sure that grafDOS is loaded into memory.

2. Make sure MINI-MON is on the disk in the drive at the present time.

3. Type BLOAD "MINI-MON" <ret> .

4. Wait for the disk drive light to go off, then type EXIT <ret>. (You can also access by SYS 49152)

5. The screen should clear, turn black, and print "COMMODORE 64 MINI-MONITOR" on the screen followed by a "<".

6. The MINI-MON is now running and ready to accept a command.

7. Before going any farther, make a copy of this program! See the appendix in the back of this section.

8. If the disk drives light does not go off after loading the MINI-MON, try turning the power to the Commodore 64 and starting again. If the light still stays on, the disk is probably bad and should be exchanged.

III. COMMAND DESCRIPTION

X COMMAND - This command displays six memory locations at a specified location in hex and text. After the X is pressed, enter a hexadecimal address up to 4 digits long. If the address is less than 4 digits, then you must press the space bar. For example, to look at the memory at location $10, use the following key sequence:

X 10 <SP>

The value of memory at locations $10-$15 will be displayed. You will see to the right of the six hexadecimal numbers the actual text characters the numbers represent.

<SPACE> COMMAND - This displays the next six memory locations in hex and ASCII text. For example, if a space is entered after the previous example, the contents of locations $16 to $1A will be displayed. By holding down the space bar, you can rapidly step through memory.

/ COMMAND - This command changes memory at the location last given by a X or another / command. For example, to write the numbers 1 thru 6 into location $100 type:

X 100 <SP> / 01 02 03 04 05 06

If you type less than 6 digits, you must type <RET> to enter the changes. The monitor will automatically return to the prompt. To continue to make changes, type another / for the next six memory locations.

D COMMAND - This command disassembles 20 lines of 6502 code. After the D key is pressed, type in a 4 digit hex address. If the address is less than 4 digits long, then the space bar must be pressed. If the D command is entered without an address, the monitor will disassemble from the last memory location.

T COMMAND - This command displays the next 30 memory locations in ASCII text format that was last accessed by either the D command or X command.
If you would like this command to auto-repeat, there is a memory location that will make the entire keyboard repeat when you hold down the key. To enable this option, type:

X 028A / FF <RET>

To change it back, put 00 instead of FF at that memory location.

G COMMAND - This command jumps to a specified location. For example, to run a program at location 1000 type:

G 1000

M COMMAND - This command moves memory over a specified range to another memory location. After typing M the computer will ask for a memory range (from=, to=) and then ask for the destination to move the memory. Type in the hexadecimal address up to 4 digits long. If it is less than 4 digits, you must press the space bar to enter the number.

H COMMAND - This command hunts for a string in memory.The memory range that is searched is from $0000 - $BFFF and $E000 - $FFFF. The string may be input in either ASCII or hex. If it does not find it, you will return back to the prompt.

F COMMAND - This command fills a section with any byte. After pressing the F key enter the fill byte and the from and to memory address.

R COMMAND - This command switches the kernal ($D000-FFFF) and BASIC ($A000 - BFFF) to RAM. The kernal and BASIC sections can then be changed.

Z COMMAND - This is the opposite of the R command, it switches the kernal and BASIC back to ROM.

? COMMAND - This command displays a one page command summary.

COMMAND - This command converts a decimal (max 65535) to a hex number. Example, convert 8192 into hex.

#8192 <SPACE>

The correct hex number will be $2000.

$ COMMAND - This is the opposite of the # command, it converts a hex number to a decimal number. This command works the same way as the # command.

²Q COMMAND - This command exits to BASIC without clearing any program in memory. This command should work most of the time but if certain memory locations are changed it could erase the BASIC program. (² = control)
To restart MINI-MON, type: SYS 49152.
To re-enter grafDOS, you must type: BASIC.

²R COMMAND - This command does a reset (JMP ($FFFC)). It will clear any BASIC program in memory, however the mini-mon can be restarted by typing SYS 49152.

W COMMAND - This command restarts the mini-mon program. It does the same as SYS 49152 from BASIC or G C000 from the monitor.

S COMMAND - This command saves any portion of memory to disk or tape. After typing S, specify a file name then the range of memory to be saved.

WARNING: If certain memory locations are changed the computer will not save the program properly and give no indication to the user. It is best to verify that your program has been saved before turning off the power.

Also, you cannot resave a program with the same file name on the disk, you must give it a new name.

L COMMAND - This command loads a file from disk or tape. After entering the file name press return and wait for the disk drive light to go off or the tape to stop turning. The file should then be loaded. The program will always load into the same memory locations from which it was saved to disk or tape.

MINI-ASSEMBLER

A COMMAND - This command calls the assembler. After pressing the A key the computer should print a "*". Instructions are entered by typing an address followed by a colon then the instruction and it's value (if any). Example, to enter the instruction LDA #$FF into location $1000 you would type:

1000:LDA #$FF <RET>

The disassembled form of the instruction will then be displayed followed on the next line by the "*". To enter an instruction at the next available location ($1002) type a space instead of 1002:. For example, to enter STA $2000 at location $1002 type:

<SPACE> STA $2000

If you type something illegal, the Mini-assembler will flag the error and refuse to enter the instruction. You must retype the instruction with the correct syntax.

To exit the mini-assembler mode, you must type Q <RET> to return back to the mini-monitor.

NOTE: Programs created with mini-mon will require you to add ' 1,1 ' or ' 8,1 ' after the filename of your program when loading from BASIC.

For example, LOAD"filename",1,1 for tape
LOAD"filename",8,1 for disk

This will insure that your program is loaded correctly into memory.

In grafDOS BASIC the command would be:

BLOAD"filename"

*** NOTE: At many of the prompts given in MINI-MON, you may abort the command by typing Control - C.

IV. BACKING UP THE MINI-MON

1. After entering the MINI-MON as described in the getting started section, take out the disk in the drive and insert the backup disk. Note - the disk must be initialized.

   To initialize a disk, type:

   ```
   OPEN 15,8,15
   PRINT #15,"NEWO:MINI MON DISK,64"
   ```

   In grafDOS BASIC, simply type:

   ```
   INIT "MINI MON DISK"
   ```

   It will take several minutes for the disk to be initialized. You must reload the MINI-MON after initializing a disk.

2. Type S followed by any file names. The address range to be saved is FROM=$C000 TO=$CFFF.

3. After the disk drive light goes off, exit to BASIC and examine the catalog to make sure the backup copy has been saved.

4. Put away the original and use this copy for your work. If anything should happen to it, you will still have the original to make a new copy.

APPENDICES

MEMORY MAP OF grafDOS BASIC

```
$0000-$03FF : ZERO-PAGE & LANGUAGE WORKING AREA
$0400-$07FF : VIDEO SCREEN
$0800-$7FFF : BASIC SYSTEM MEMORY
$8000-$8FFF : GRAFDOS BASIC
$9000-$9FFF : HIRES/PROGRAMMABLE CHARACTERS AREA
$A000-$BFFF : COMMODORE 64 BASIC ROUTINES (USED BY GRAFDOS)
$C000-$CFFF : MINIMON (IF PRESENT)
$D000-$FFFF : OPERATING SYSTEM / HARDWARE
```

ERROR CODES & THEIR NUMBERS

The following errors are PEEKed after a TRAP statement has been utilized by grafDOS BASIC

PEEK(2)	ERROR
1	TOO MANY FILES
2	FILE ALREADY OPENED
3	FILE NOT PREVIOUSLY OPENED
4	FILE NOT FOUND
5	DEVICE NOT PRESENT
6	NOT INPUT FILE
7	NOT OUTPUT FILE
8	MISSING FILE NAME
9	ILLEGAL DEVICE NUMBER
10	NEXT WITHOUT FOR
11	SYNTAX
12	RETURN WITHOUT GOSUB
13	OUT OF DATA
14	ILLEGAL QUANTITY
15	OVERFLOW
16	OUT OF MEMORY
17	UNDEF'D STATMENT
18	BAD SUBSCRIPT
19	REDIM'D ARRAY
20	DIVISION BY ZERO
21	ILLEGAL DIRECT
22	TYPE MISMATCH
23	STRING TOO LONG
24	FILE DATA
25	FORMULA TOO COMPLEX
26	CAN'T CONTINUE
27	UNDEF'D FUNCTION
28	MISSING FILE SEPARATOR
29	DOS SYNTAX
30	WRITE PROTECTION
31	DISK
32	UNDEF'D BASIC COMMAND (not utilized)
33	LANGUAGE NOT AVAILABLE
34	MEMORY PROTECTION

LIST OF PEEKs AND POKEs

Although there are really only a few workspace pointers that the BASIC programmer could utilize, we have chose to list them here so that machine language writers will know which memory locations are utilized during normal language operation.

```
$0002 : ERROR # FROM LAST TRAP COMMAND
$0090 : CURRENT DISK STATUS BYTE
$033C : START OF FILE NAME BUFFER #1
$035C : START OF FILE NAME BUFFER #2
$037C : START OF MISC FILE BUFFER #3

34174 - chr used in lowres plot (usually 160-a white square)
34507 - low frequency for CTRL/G
34512 - high frequency for CTRL/G
34517 - duration for CTRL/G
35115 - attack/decay for all sound
35120 - waveform for all sound
35125 - volume for all sound
35851 - low frequency for error "beep"
35856 - high frequency for error "beep"
35861 - duration for error "beep"
```

CASSETTE LOAD AND SAVE

Some of you may ask about cassette loads and saves. Since LOAD and SAVE behave differently under grafDOS BASIC, you must use a different command to access the tape. The syntax for cassette operation is as follows:

```
AOAD "filename"      (load)

AAVE "filename"      (save)
```

MAKING BACKUP COPIES OF grafDOS BASIC

1. Make sure that the grafDOS BASIC language is up & running.
2. Insert your MASTER DISK, then type: LOAD "GRAFDOS"
3. Put an initialized disk into the drive.
4. Type: SAVE "GRAFDOS"
5. Now, type: BSAVE "GRAFDOS.SOURCE",32768,36864
6. This will SAVE the language to your new diskette!
7. You have now successfully "backed-up" your system disk.
8. Check the disk by turning off the system and reloading.

SUMMARY OF
grafDOS Disk Operating System Commands

INIT "disk name"

This command initializes the disk currently in Drive 0 under the name "disk name". Initializing a disk erases all previously store information on that disk. Before this command is executed, grafDOS prompts you with ARE YOU SURE?... Type 'Y' if you're sure, otherwise it will not initialize the disk.

```
INIT "GRAFDOS MASTER DISK"
```

...will initialize the disk in Drive 0 and call it "GRAFDOS MASTER DISK".

UPDATE

This command UPDATEs your disk so that all free sectors are placed in consecutive order.

* Note - This command is very dangerous if your are currently using sequential files on your diskette. This command will destroy pieces of the files, so do not use this command if any sequential files are found on your disk!!

SAVE "file name"

SAVEs the current program in memory under the name "file name". If a program already exists on the disk under this file name, it will be replaced by your program in memory.

```
SAVE "MY PROGRAM!"
```

...will save your current program in memory on disk.

LOAD "file name"

LOADs the BASIC program called "file name" from your disk, it will erase any program that is presently in memory before this command was executed.

```
LOAD "A PROGRAM"
```

...will load "A PROGRAM" off the disk.

CATALOG

Lists the diskette catalog (or directory) of Drive 0. An average directory might look something like this:

DISKETTE DIRECTORY: GRAFDOS MASTER DISKETTE

FILE NAME	TYP	BLOCKS
GRAFDOS	PRG	1
GRAFDOS.SOURCE	PRG	17
MY PROGRAM	PRG	1

1. The "GRAFDOS MASTER DISKETTE" in the first line is the name of the disk.

2. The FILE NAME column lists the file names of the programs that you have currently SAVEd on your disk.

3. The TYP column lists what type of file that it is, the file types are as follows:

   PRG ... BASIC program
   SEQ ... Sequential file
   USR ... User

4. The BLOCKS column lists how much disk space the file takes up. If you want to know how much memory the file uses, simply multiply the number of blocks by 256.

DELETE "file name"

This command DELETEs the file called "file name" from your disk, also erasing it from your diskette directory. You WILL NOT be able to get the program back after this command so use it with care.

```
DELETE "AN UNWANTED PROGRAM"
```

...removes the file "AN UNWANTED PROGRAM" from your diskette.

RENAME "old name","new name"

RENAMEs the file called "old name" to "new name". This command is useful for changing the names of the programs on your diskette.

```
RENAME "AN OLD ONE","A NEW ONE!"
```

...RENAMEs the file called "AN OLD ONE" to "A NEW ONE!".

* Note - Be very careful with this command! For example:

1. Let's say that the program "PROGRAM" exists on your disk.

2. Now you type RENAME "OLD PROGRAM","PROGRAM"...

   ...you will find TWO programs called "PROGRAM" on your disk!

3. This command does not check for identical filenames so be sure that the new name is not already on the disk.

BASIC

This command restarts (re-initializes) the grafDOS BASIC system. Also, it will not destroy your program currently in memory. All it simply does is to rehook grafDOS BASIC into memory.

This command is primarily used when RUNSTOP/RESTORE has been pressed.

EXIT

EXITs from BASIC into the machine language monitor, MINIMON. This command will only work if MINIMON is present in memory, otherwise it will return ?LANGUAGE NOT AVAILABLE ERROR. To load in the minimon, type in the following:

```
BLOAD "MINIMON"
```

BSAVE "file name",aexpr1,aexpr2

SAVEs the block of memory from /aexpr1/ to /aexpr2/ with the name "file name". /aexpr1/ and /aexpr2/ must be VALID memory locations, otherwise this command will fail!

```
BSAVE "SCREEN",1024,2039
```

...will save the video screen to disk, calling it "SCREEN"...

BLOAD "file name"

LOADs the memory file of a BSAVEd program. The file will be loaded into the same memory locations when it was saved.

```
BLOAD "SCREEN"
```

...will load back in the video screen off disk... (see BSAVE)

NOTE: New CBM-64s must poke the screen a different color because of new revisions in the video chip. (POKE 53281,6)

STAT "MEM" and STAT "SYS"

Will give you the following system statistics:

1. If command="MEM", STAT will perform memory diagnostics.
2. If command="SYS", STAT will perform system diagnostics.

RUN "filename"

Instead of loading and running a program, a much faster and simpler way can b accomplished by using this command. By typing:

```
RUN "MY PROGRAM"
```

will automatically load and run the program in one step.

WATCH and OFF

No parameters. DOS commands do not normally get printed to the screen. However, sometimes it is helpful to see what is happening. The WATCH command will allow just that by printing to the screen any disk functions. Conversely, the OFF command will turn off this operation.

***** NOTE: For all of the grafDOS command set, it is not necessary to type in the entire command since grafDOS only checks for the first two letters of the command word.

SUMMARY OF grafDOS BASIC commands:

TEXT

No parameters. Sets the video screen to normal text/graphics mode (25x40) from either low or high resolution graphics mode. The video screen is also cleared by this command.

```
10 REM SET NORMAL TEXT MODE
20 TEXT
```

LGR

No parameters. Sets the video screen to low resolution mode, creating a color graphics display screen resolution of (25x40), in 16 colors.

```
10 LGR:REM SET LORES GRAPHICS MODE
```

LCOL aexpr

Sets the color for low resolution plotting. The range for /aexpr/ is from 0-255, but the color is treated modulo 16. The color value are as follows:

0 Black	7 Yellow	14 Light Blue
1 White	8 Orange	15 Gray 3
2 Red	9 Brown	
3 Cyan	10 Light Red	
4 Purple	11 Gray 1	
5 Green	12 Gray 2	
6 Blue	13 Light Green	

```
10 LCOL 6:LPLOT 10,10:REM PLOT BLUE POINT AT (10,10)
```

LPLOT aexpr1,aexpr2

This command plots a low resolution point, with the x-coordinate /aexpr1/, and y-coordinate /aexpr2/. The color of the point is determined by the last LCOL command. The origin for all graphics (0,0) is at the upper left corner of the video screen.

```
10 LPLOT 0,0:REM PLOT POINT AT (0,0)
```

HLIN aexpr1,aexpr2 @ aexpr3

Draws a horizontal line between the points (aexpr1,aexpr3) and (aexpr2,aexpr3). It uses the color previously set by the LCOL command.

```
10 HLIN 0,39 @ 10:REM DRAWS HORIZONTAL LINE FROM (0,10) - (39,10)
```

VLIN aexpr1,aexpr2 @ aexpr3

Same as HLIN, but draws a vertical line between the points (aexpr3,aexpr1), and (aexpr3,aexpr2).

```
10 VLIN 0,20 @ 10:REM DRAWS VERTICAL LINE FROM (10,0) - (10,20)
```

HGR

No parameters. Sets the computer to high resolution graphics mode, clearing the hires screen to black. The resolution of the screen is (255x192), allowing vividly detailed high resolution pictures to be drawn.

* Note - A very long program (beyond 8k) will be destroyed by the HGR command, since it clears that 8k of memory. To protect against this, add the HIMEM command to your BASIC program, as follows:

1. For programs shorter than 8k:

```
100 HGR:REM CLEAR HIRES SCREEN
```

2. For programs beyond 8k:

```
100 HIMEM 8192
110 HGR
```

PLOT aexpr1,aexpr2

Plots a high resolution point with an x-coordinate of /aexpr1/, and a y-coordinate of /aexpr2/.

```
100 PLOT 100,95:REM PLOT HIRES POINT AT (100,95)
```

SCREEN aexpr

Changes the color of all points on the high resolution screen to the color /aexpr/. Usually, all points will be plotted in white. Color range is from 0 to 15.

```
100 SCREEN 2:REM CHANGE ALL HIRES POINTS TO RED
```

FLIP

No parameters. Flips from the text screen to the high resolution screen without clearing the high resolution screen first.

```
100 FLIP:REM HERE WE GO TO THE HIRES SCREEN!!
```

DRAW aexpr1,aexpr2,aexpr3

Draws the sprite number /aexpr1/, at the high resolution point (aexpr2,aexpr3). The sprite must have already been enabled before this command is used, otherwise no change will appear on the video screen. For a more detailed look at sprites, consult your users manual from Commodore.

```
100 DRAW 7,100,100:REM DRAW SPRITE #7 AT (100,100)
```

KEY

No parameters. This command waits for something to be typed on the keyboard. The character typed can be retrieved by PEEKing location 631, as follows:

1. To just wait for a keypress:

```
100 KEY:REM LET'S WAIT FOR A KEY...
```

2. To retrieve the key pressed:

```
100 KEY:C=PEEK(631):REM C WILL CONTAIN THE ASCII
                       VALUE OF THE CHARACTER.
```

WCHAR aexpr1,aexpr2 @ aexpr3

Writes at high resolution coordinate (aexpr1,aexpr2), the character number /aexpr3/. For example, if /aexpr3/ = 1, then an 'A' would be plotted at that point.

Note - See the commands ALT,COPY, and NOM for detailing how to use the WCHAR command more fully.

```
100 WCHAR 10,10 @ 1:REM PLOT "A" AT (10,10) ON HIRES SCREEN
```

SOUND aexpr1,aexpr2 @ aexpr3

Sounds the tone (from voice one only), with the high-frequency value of /aexpr1/, and low frequency value of /aexpr2/, for a duration of /aexpr3/. The higher /asexpr3/, the longer the tone will be. See the users manual for musical values.

```
100 SOUND 8,147 @ 200:REM SOUND C (OCTAVE 3) FOR DELAY OF 200
```

ALT

No parameters. Specifies that you wish to use your own character set for the WCHAR command, rather than using Commodore's character set. The character set definition will begin at $9000, and will have the same data syntax as all Commodore character sets. (See CBM-64 Reference manual for designing your own characters)

```
100 ALT:REM I WANT MY OWN CHARACTER SET.
```

COPY

No parameters. This command copies the Commodore character set into your own table ($9000), allowing you to be able to modify certain characters without having to define them all. Use this command after the ALT command.

```
100 ALT :USE OWN CHARACTER SET.
110 COPY:REM USE COMMODORE'S CHARACTER SET TO BE MODIFIED.
```

NOM

No parameters. Specifies that you wish to use Commodore's character set, and do not wish to use your own characters. This is default for the WCHAR command.

```
100 NOM:REM SET FOR NOrMal CHARACTER SET
```

PIC "string"

Loads the high resolution picture stored on disk called "string". If high resolution graphics have not already been enabled, this command will have no visible effect on the video screen. /string/ is not a true string, in the sense that it cannot be replaced with A$. But, PIC "PICTURE" will work.

```
Correct syntax:     100 PIC "MY.LOGO"
Incorrect syntax:   100 A$="MY.LOGO":PIC A$
```

PSAVE "string"

Saves the current high resolution screen to disk, calling it "string". The same restrictions placed on "string" in PIC apply for this command also.

```
100 PSAVE "MY PICTURE":REM SAVE HIRES SCREEN AS "MY PICTURE"
```

HOME

No parameters. Places the cursor at the top of the video screen, clearing the video screen at the same time.

```
100 HOME:REM CLEAR VIDEO SCREEN, PLACE CURSOR AT TOP
```

VTAB aexpr

Moves the cursor to the video line /aexpr/. For example, VTAB 0 would place the cursor on the top line, where VTAB 10 would move the cursor to the middle of the screen (line #10). Its range is from 0 to 24.

```
100 VTAB 15:REM HERE WE GO DOWN TO LINE 15!
```

HTAB aexpr

Moves the cursor to the column /aexpr/. Functions the same as TAB(X), but is not used within a PRINT statement. Unlike TAB(X), it can also move back through the screen line. Its range is from 0 to 39.

```
100 HTAB 20:REM MOVE TO MIDDLE OF SCREEN
```

HIMEM aexpr

Sets the upper memory limit to /aexpr/. This allows the user to "protect" the upper memory from being destroyed by the program in memory, for example, the high resolution screen can be protected in this manner. Remember to reset it back to 32767.

```
100 HIMEM 8192:REM LET'S PROTECT THE HIRES SCREEN!
```

CHAIN "string"

LOADs & RUNs the file called "string". This command does not save variable definitions, but simply "chains" two programs together. The same restrictions to "string" apply here too!

```
100 CHAIN "PART TWO":REM RUN PROGRAM CALLED "PART TWO"
```

SPEED aexpr

This command sets the printing speed (video only) to delay /aexpr/ number of times. The larger /aexpr/ is, the greater the delay becomes. Its range is from 0 to 255.

```
100 SPEED 255:REM DELAY A LONG TIME!!!
```

TRAP aexpr

When an error occurs from within a program, the program does not abort, but transfers control to line number /aexpr/, which can then handle the error... If the programmer wishes to know which error occurred, the error number can be retrieved by PEEKing memory location 2.

```
100 TRAP 1000:REM GOTO LINE 1000 IF AN ERROR OCCURS
```

For example:

```
100 TRAP 1000
110 PRINT 0/0
120 END
1000 PRINT "ERROR CODE #";PEEK(2)
```

When you RUN this program, the #20 will be PRINTed, for that is the code for DIVISION BY ZERO ERROR...

INDEX